La Salle University
La Salle University Digital Commons

Mathematics and Computer Science Capstones | Mathematics and Computer Science, Department of

Spring 5-19-2016

FINDAPark Implementation

David Laratta
La Salle University, laratta@student.lasalle.edu

Follow this and additional works at: http://digitalcommons.lasalle.edu/mathcompcapstones

 Part of the Other Computer Sciences Commons, and the Programming Languages and Compilers Commons

Recommended Citation
Laratta, David, "FINDAPark Implementation" (2016). *Mathematics and Computer Science Capstones*. 29.
http://digitalcommons.lasalle.edu/mathcompcapstones/29

This Thesis is brought to you for free and open access by the Mathematics and Computer Science, Department of at La Salle University Digital Commons. It has been accepted for inclusion in Mathematics and Computer Science Capstones by an authorized administrator of La Salle University Digital Commons. For more information, please contact careyc@lasalle.edu.

CIS Capstone

David Laratta

4/17/16

Table of Contents

David Laratta

System Proposal

Executive Summary

System Request

FindAPark is a web service application designed to help a user search for a local hike or vacation revolving around hiking or cycling.

Workplan

The workplan for this capstone is to create the System Proposal which includes detailed information about the completion of all the items in the System Proposal except the Functional, Structural, and Behavioral Models.

Requirements Definition

Describes the Functional and Non-Functional Requirements. The Non-Functional Requirements consist of Operational, Performance, Security, and Cultural or Political Requirements. The FindAPark system will run on any system with an internet connection using Internet Explorer, Mozilla Firefox, and Google Chrome. Other browsers might work, but will not be tested.

The Functional Requirements describe the functionality of the system and include the downloading and installing of the application, the user profile, the workout routine, the transfer of data, and the review of workout data as well as customized workouts.

Feasibility Analysis

The feasibility describes the technical feasibility of the Fitness Wire system. It has concluded that the project is feasible technically with some risk. Potential risks include familiarity with fitness applications, familiarity with the technology used to develop the application, and the project size.

FindAPark is suited for anyone who is interested in hiking, trail riding, and walking. The website has been designed to allow people to search for these locations using Google Maps. FindAPark displays parks based off of the users selected location. Hiking, walking, and trail riding is suited for a wide range of ages (athletes, middle aged men and women, individuals who are overweight, and the elderly) and backgrounds as shown in the personas below.

Personas

Jeff

Jeff is an average working male in his early-50's looking to stay in shape through the use FindAPark. He enjoys occasional hikes and weekly bike rides through local parks and even takes yearly trips to Acadia National Park with his family. FindAPark offers the ability to search for local parks to hike and ride in. FindAPark also provides the ability to search for National Parks when searching for the city that it is located in.

Kris

Kris is in his mid-20s and is a frequent hiker. He enjoys traveling around the United States finding new trails to hike with every trip. With FindAPark Kris will be able to search for parks, read information and view pictures of the areas that intrigue his interest.

Dan

Dan is collegiate all-star cyclist who enjoys daily rides. Each week he looks for new rides to experience different sceneries. FindAPark will allow Dan to search for parks that have beautiful scenery using the Instagram photo plug-in.

Anderson Family

The Anderson family enjoys taking monthly family trips to local parks to hike and cycle. Since their family ranges in age, they like to find parks that are family oriented and that have great reviews. FindAPark provides all these features to help the Anderson family in search for a park that meets their needs.

System Request – FindAPark

Capstone Advisor: Dr. Blum

Business Need: This capstone has been designed to help users search for a local hike or vacation revolving around hiking or cycling.

Business Requirements:

Using the web, users will be able to access FindAPark. The functionality the system will have is as follows:

- Be able to access the Internet from any platform with internet connectivity.
- Be able to search for parks using the Google Map Plug-in.

Business Value:

I expect this service will improve on people finding information about park significantly fasters.

Special Issues or Constraints:

- The Web Application should be on the Internet by April 2016.

Work Plan

Work Break down Structure

Task Number	Task Name	Duration	Status
1	Develop work plan		Complete
1.1	Review Work plan		Complete
2	Create Standards		Complete
2.1	Review Standards		Complete
3	System Request		Complete
3.1	Review System Request		Complete
4	Feasibility Analysis		Complete
4.1	Feasibility Analysis review		Complete
5	Requirements Definition		Complete
5.1	Requirements Definition review		Complete
6	Project Charter		Complete
6.1	Review Project Charter		Complete
7	Executive Summary		Complete
7.1	Executive Summary review		Complete
8	Create Table of Content		Complete

8.1	Review table of content		Complete
9	Develop an Activity Diagram		Complete
9.1	Review Activity Diagram		Complete
10	Develop Use Cases		Complete
10.1	Review Use Cases		Complete
11	Develop Use Case Diagram		Complete
11.1	Review Use Case Diagram		Complete
12	Create Annotated Wireframes		Complete
12.1	Review Annotated Wireframes		Complete
13	Create Verification and Validation Report		Complete
13.1	Review Verification and Validation Report		Complete
14	Write Code		Complete
14.1	Review Code		Complete
15	Create Challenges Encounters		Complete
15.1	Review Challenges Encounters		Complete
16	Create Lessons Learned		Complete
16.1	Review Lessons Learned		Complete
17	Create Suggestions for Additional Research		Complete
17.1	Review Suggestions for Additional Research		Complete
18	Update Work plan		Complete
18.1	Review Updated Work plan		Complete

Work plan Information

System Proposal

This work plan breaks down the hours spent on designing, documenting and coding. Majority of the time spent on this project was developing (coding) the actual web application.

1. Develop work plan
    a. Start Date : 10/24/11
    b. Completion date : 10/25/11
    c. Deliverables : Work plan
    d. Completion status: complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 2 hours
    h. Actual time: 1.5 hours
2. Create Standards
    a. Start Date : 10/23/11
    b. Completion date : 10/23/11
    c. Deliverables : Team Standards Document
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 2.5 Hours
3. System Request
    a. Start Date : 10/19/11
    b. Completion date : 10/23/11
    c. Deliverables : System Request Document
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 1.5 Hours
4. Feasibility Analysis
    a. Start Date : 10/19/11
    b. Completion date : 10/23/11
    c. Deliverables : Feasibility Analysis Document
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 1.5 Hours
5. Requirements Definition
    a. Start Date : 10/19/11
    b. Completion date : 10/25/11

   c. Deliverables : Requirements Definition Document
   d. Completion status: Complete
   e. Priority: High
   f. Resources needed: Word processing software
   g. Estimated time : 3 Hours
   h. Actual time: 2.5 Hours
6. Project Charter
   a. Start Date : 10/22/15
   b. Completion date : 10/22/15
   c. Deliverables : Project Charter Document
   d. Completion status: Complete
   e. Priority: High
   f. Resources needed: Word processing software
   g. Estimated time : 3 Hours
   h. Actual time: 1 Hour
7. Executive Summary
   a. Start Date : 10/23/15
   b. Completion date : 10/25/15
   c. Deliverables : Executive Summary Document
   d. Completion status: Complete
   e. Priority: High
   f. Resources needed: Word processing software
   g. Estimated time : 3 Hours
   h. Actual time: 1 Hour
8. Table of Contents
   a. Start Date : 10/23/15
   b. Completion date : 10/25/15
   c. Deliverables : Table of Contents
   d. Completion status: Complete
   e. Priority: High
   f. Resources needed: Word processing software
   g. Estimated time : .5 Hours
   h. Actual time: <.5 Hours
9. Activity Diagram
   a. Start Date : 10/30/15
   b. Completion date : 11/1/15
   c. Deliverables : Activity Diagram
   d. Completion status: Complete
   e. Priority: High
   f. Resources needed: Visio
   g. Estimated time : 1 Hour
   h. Actual time: 1 Hour

10. Use Cases
    a. Start Date : 10/30/15
    b. Completion date : 11/8/15
    c. Deliverables : Set of Use Cases
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 3 Hours
11. Use Case Diagram
    a. Start Date : 10/30/15
    b. Completion date : 11/8/15
    c. Deliverables : Use Case Diagram
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Visio
    g. Estimated time : 1 Hours
    h. Actual time: 1 Hour
12. Annotated Wireframes
    a. Start Date : 4/19/16
    b. Completion date : 4/19/16
    c. Deliverables : Wireframes
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Visio and Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 2 Hour
13. Verification and Validation Report
    a. Start Date : 11/8/15
    b. Completion date : 11/8/15
    c. Deliverables : Verification and Validation report
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 80 Hours
    h. Actual time: 100 Hours
14. Coding
    a. Start Date : 11/8/15
    b. Completion date : 3/6/16
    c. Deliverables : Code
    d. Completion status: Complete
    e. Priority: High

f. Resources needed: Development Software
g. Estimated time : 80 Hours
h. Actual time: 132 Hours

15. Challenges Encounters
    a. Start Date : 4/17/16
    b. Completion date : 4/17/16
    c. Deliverables : Challenges Encounters
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 3 Hours
16. Lessons Learned
    a. Start Date : 4/17/16
    b. Completion date : 4/17/16
    c. Deliverables : Lessons Learned
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 3 Hours
    h. Actual time: 2 Hours
17. Suggestions for Additional Research
    a. Start Date : 4/17/16
    b. Completion date : 4/17/16
    c. Deliverables : Suggestions for Additional Research
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 2 Hours
    h. Actual time: 1 Hours
18. Update Work plan
    a. Start Date : 2/14/16
    b. Completion date :4/17/16
    c. Deliverables : Updated Work plan
    d. Completion status: Complete
    e. Priority: High
    f. Resources needed: Word processing software
    g. Estimated time : 4 Hours
    h. Actual time: 3 Hour

Requirements Definition

Nonfunctional Requirements

1. Operational Requirements
   1.1. FindAPark will run on Internet Explorer 8+, Firefox 42+, and Google Chrome 42+.
   1.2. FindAPark will use a Google Maps Plug-in, so you system must meet these system requirements.
      1.2.1. **Requirements for Windows**
         1.2.1.1. Operating System: Windows XP
         1.2.1.2. CPU: Pentium 4 2.4GHz+ or AMD 2400xp+
         1.2.1.3. System Memory (RAM): 512MB
         1.2.1.4. Hard Disk: 2GB free space
         1.2.1.5. Network Speed: 768 Kbits/sec
         1.2.1.6. Graphics Card: 3D-capable with 32MB of VRAM
         1.2.1.7. Screen: 1280x1024, "32-bit True Color"
      1.2.2. **Requirements for Mac OS X**
         1.2.2.1. Operating System: Mac OS X 10.4.5
         1.2.2.2. CPU: G4 1.2Ghz
         1.2.2.3. System Memory (RAM): 512MB
         1.2.2.4. Hard Disk: 2GB free space
         1.2.2.5. Network Speed: 768 Kbits/sec
         1.2.2.6. Graphics Card: 3D-capable with 32MB of VRAM
         1.2.2.7. Screen: 1280x1024, "Millions of Colors"
      1.2.3. **Requirements for Linux**
         1.2.3.1. Kernel 2.6 or later
         1.2.3.2. glibc 2.3.5 w/ NPTL or later
         1.2.3.3. x.org R6.7 or later
         1.2.3.4. System Memory (RAM): 512MB
         1.2.3.5. Hard Disk: 2GB free space
         1.2.3.6. Network Speed: 768 Kbits/sec
         1.2.3.7. Graphics Card: 3D-capable with 32MB of VRAM
         1.2.3.8. Screen: 1280x1024, 32 bit color
2. Performance Requirements
   2.1. The user can access the website at any time.
3. Security Requirements

3.1. All internet devices should have sufficient security for the application.
4. Cultural and Political Requirements
   4.1. TBD

Functional Requirements

1. FindAPark can be accessed by web browser
2. Home
   2.1. The user can view rotational photos of different national and state parks
   2.2. The user can search for parks, and city locations
3. Map
   3.1. The FindAPark system will work with Google Maps API
   3.2. The user can search the map for city locations
4. Parks
   4.1. The user can read about different parks
   4.2. The user can access the reviews page
   4.3. The user can view photos from Instagram of the different parks
5. Reviews
   5.1. The user can read reviews that are pulled from Google about the park they have selected
6. Help
   6.1. The user will be able to read helpful information about the FindAPark website
7. About
   7.1. The user will be able to read information about the developer and the FindAPark website

FindAPark Feasibility Analysis

Technical Feasibility

The FindAPark is feasible technically, although there is some risk.

FindAPark's risk regarding familiarity with hiking sites is low

- The developer is familiar with hiking around the east and west coast.

FindAPark's risk regarding familiarity with the technology is low

- FindAPark will be developed on Windows using the Dreamweaver IDE and in HTML, JQuery and JavaScript.
- The developer has a windows system with Dreamweaver CS4.
- The developer has experience in programming in HTML, JQuery, and JavaScript from his previous to Master courses.

- The developer is comfortable with developing on a Windows System and with HTML, JQuery, and JavaScript.

The project size is considered medium risk

- The project team consists of one person.
- The project size is relatively large, but the developer will be developing a simplex version of FindAPark, which uses a lot of outside sources. Mainly Google Maps API and Instagram plug-in.
- The project has deadlines that the developer has to meet and it has to be completed within a certain timeframe.

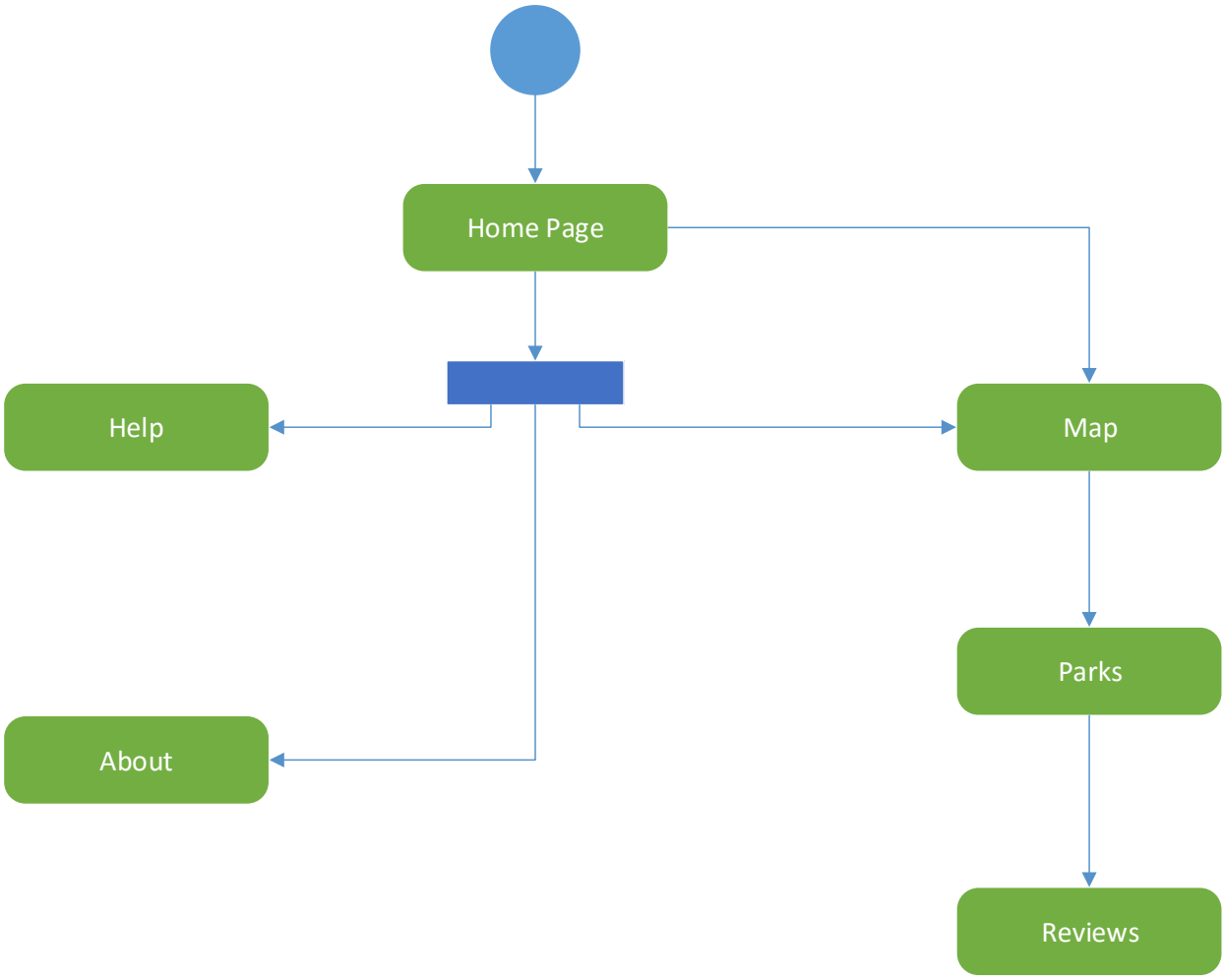
Project Chart

Project objective: Create a Web application to help users find information about parks easier.

I will:

1. Have a meeting bi-weekly on Monday at 7:00 PM EST, to report on the status and ask question with my Advisor.
2. Update the work plan with actual data each Sunday by 12:00 AM EST.
3. Discuss all problems with Dr. Thomas Blum, the advisor, as soon as they are detected.

Activity Diagram



Use Case Descriptions

Use Case Name: Home Page	ID: 1	Importance Level: High
Primary Actor: User	Use Case Type: Detailed, Essential	
Stakeholders and Interests: Users – Can search for location and parks.		
Brief Description: The home page is FindAParks first page in the website startup which I use to allow users to search for parks and locations that interest them.		
Trigger: User opens web application to Home Page. Type: External		

Relationships: Association: User Include: Extend: Generalization:
Normal Flow of Events: 1. User opens web application which starts at the home page 2. User can read home page updates 3. User can search for city 3.1. User types in a city in search text box 3.2. User will select the search button after typing in or selecting a city 4. User will be transfers to the Map page
SubFlows: None
Alternate/Exceptional Flows:

System Proposal

Use Case Name: Map	ID: 2	Importance Level: High
Primary Actor: User	Use Case Type: Detailed, Essential	

Stakeholders and Interests:

Users – Wants to view parks related to their search location from the home page or the current page.

Brief Description: The map is the second page in the FindAPark website. This page uses the Google Map API to display locations and parks based off of the users search.

Trigger: User enters a city on the home page or the user searches for a city once they navigate to the Map's page.

Type: External

Relationships:

Association: User

Include:

Extend:

Generalization:

Normal Flow of Events:

1. User accesses Map Page
2. User can search for city
   2.1. User types in a city in search text box
   2.2 a. Press enter after typing in a city
   2.2 b. User select city in dropdown box
3. User select a location on the map

SubFlows:

1. User accesses Map Page
   1.1 A. User navigates to map page through the navigation panel

   OR

   1.1 B. User is transfers to map after searching for a city on the home page

Alternate/Exceptional Flows: 1. User can use google maps to navigate around the globe

Use Case Name: Parks	ID: 3	Importance Level: High
Primary Actor: User	Use Case Type: Detailed, Essential	
Stakeholders and Interests: Users –Selects a location on the map page once they type in a city and wants to read more about the parks located in the area.		
Brief Description: This page displays several different parks located in the area and information about each park.		
Trigger: This page can only be access once the user has a city displaying in the map page. This page displays several different parks located in the area. Type: External		
Relationships: Association: User Include: Extend: Generalization:		
Normal Flow of Events: 1. User accesses Parks page once a city is selected on the map 2. User navigates to Parks page by selecting the Parks navigation tab 3. User views different parks in the location		

4. User can read about each park located in the area 5. User can select a park and view photo from Instagram of the park
SubFlows: None
Alternate/Exceptional Flows: None

Use Case Name: Reviews	ID: 4	Importance Level: High
Primary Actor: User	Use Case Type: Detailed, Essential	
Stakeholders and Interests: Users – Wants to view reviews about the park they selected.		
Brief Description: This page displays reviews, which are gathered from Google about the selected park from the parks or map page.		
Trigger: User selects the parks name inside the parks box on the parks page or selects the park on the map page and then navigates to the reviews page. Type: External		
Relationships: Association: User Include: Extend: Generalization:		
Normal Flow of Events: 1. User then will be able to access Reviews page once a city is selected on the map		

2. User will select a parks on the map 3. User will navigate to the Reviews Page by clicking the Reviews navigation tab 4. User will be able to read reviews from Google
SubFlows: 1. User will access this page after selecting the Parks name on the Parks page 2. User is transferred to the Reviews page
Alternate/Exceptional Flows:

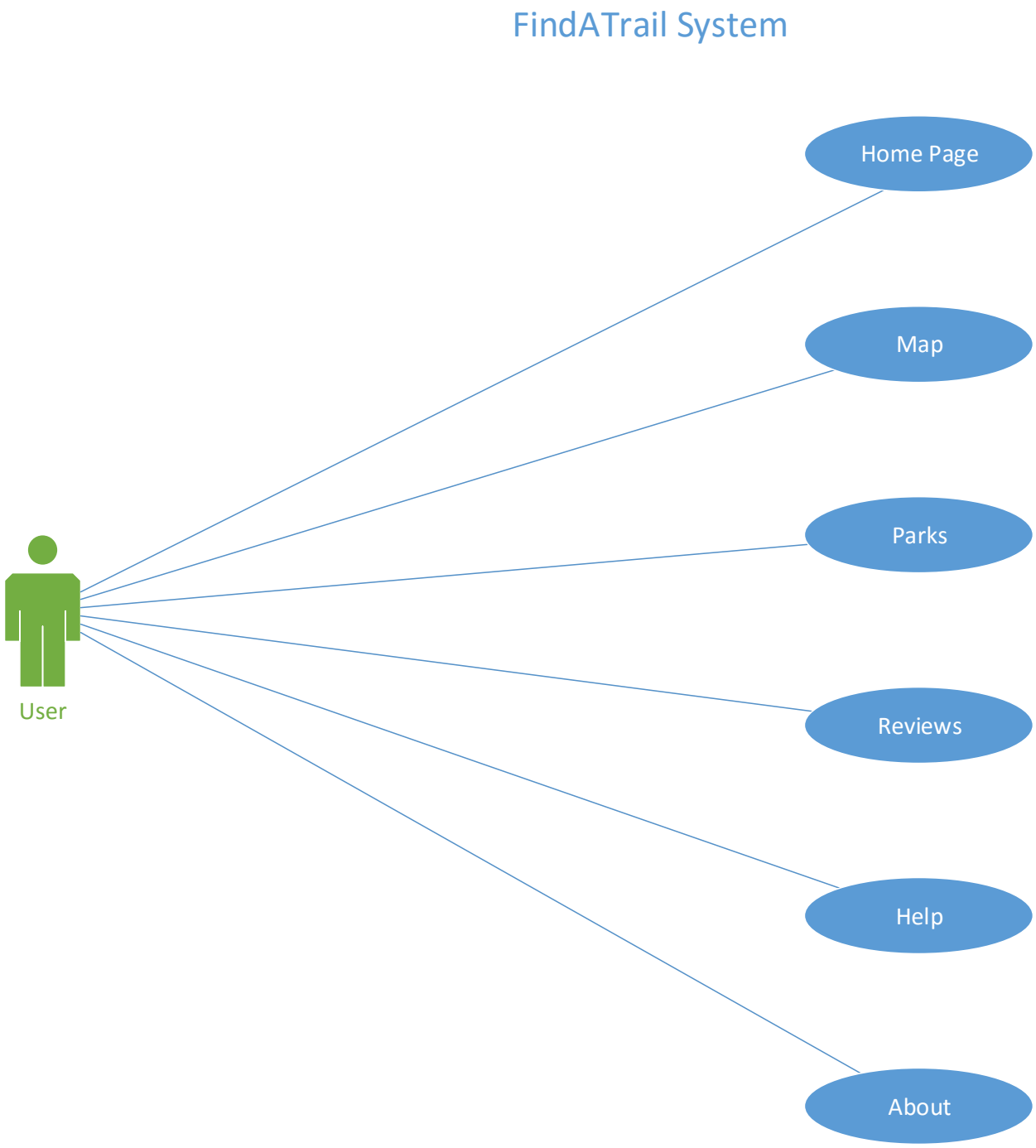
Use Case Name: Help	ID: 5	Importance Level: Low
Primary Actor: User	Use Case Type: Detailed, Essential	
Stakeholders and Interests: Users – Wants to view helpful information about FindAPark.		
Brief Description: This page displays helpful information about FindAPark.		
Trigger: User selects Help link on the top right corner of the webpage. Type: External		
Relationships: Association: User Include: Extend: Generalization:		
Normal Flow of Events: 1. Select Help Icon 2. Displays helpful information		
SubFlows: 1. Displays information about the home page		

2. Displays information about the map page 3. Displays information about the park page 4. Displays information about the review page
Alternate/Exceptional Flows: None

Use Case Name: About	ID: 6	Importance Level: Low
Primary Actor: User	Use Case Type: Detailed, Essential	
Stakeholders and Interests: Users – Wants to read about the website and user.		
Brief Description: This page displays information about the developer and the website.		
Trigger: User selects about link on top right corner of the screen. Type: External		
Relationships: Association: User Include: Extend: Generalization:		
Normal Flow of Events: 1. Select About icon 2. User will read information about the developer and FindAPark		

SubFlows: None
Alternate/Exceptional Flows: None

Use Case Diagram

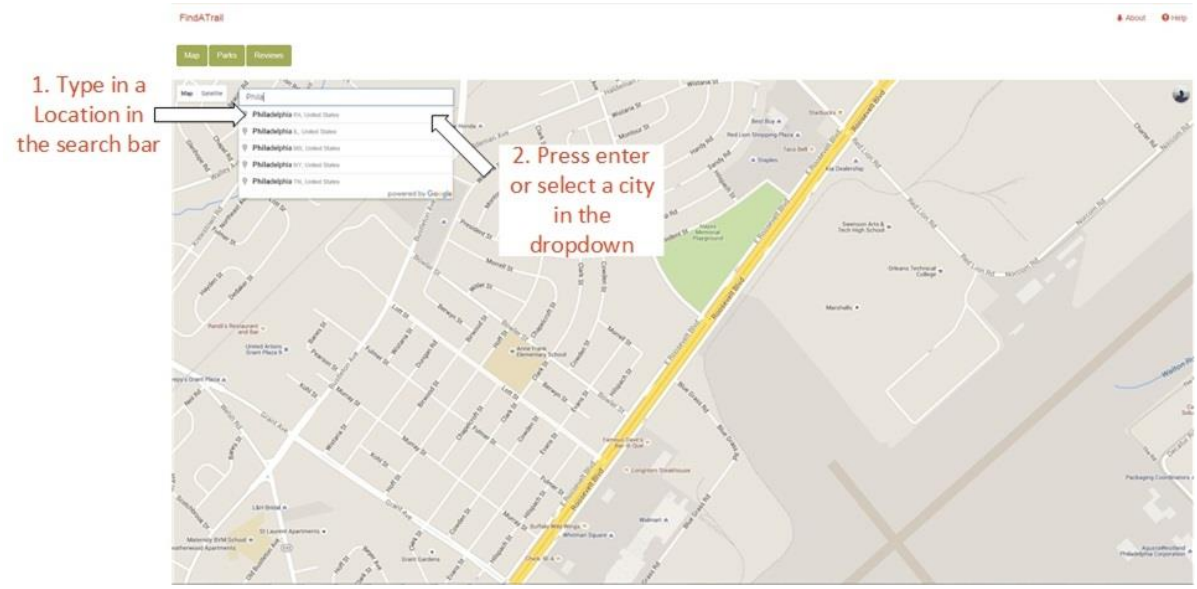


Annotated Wireframes

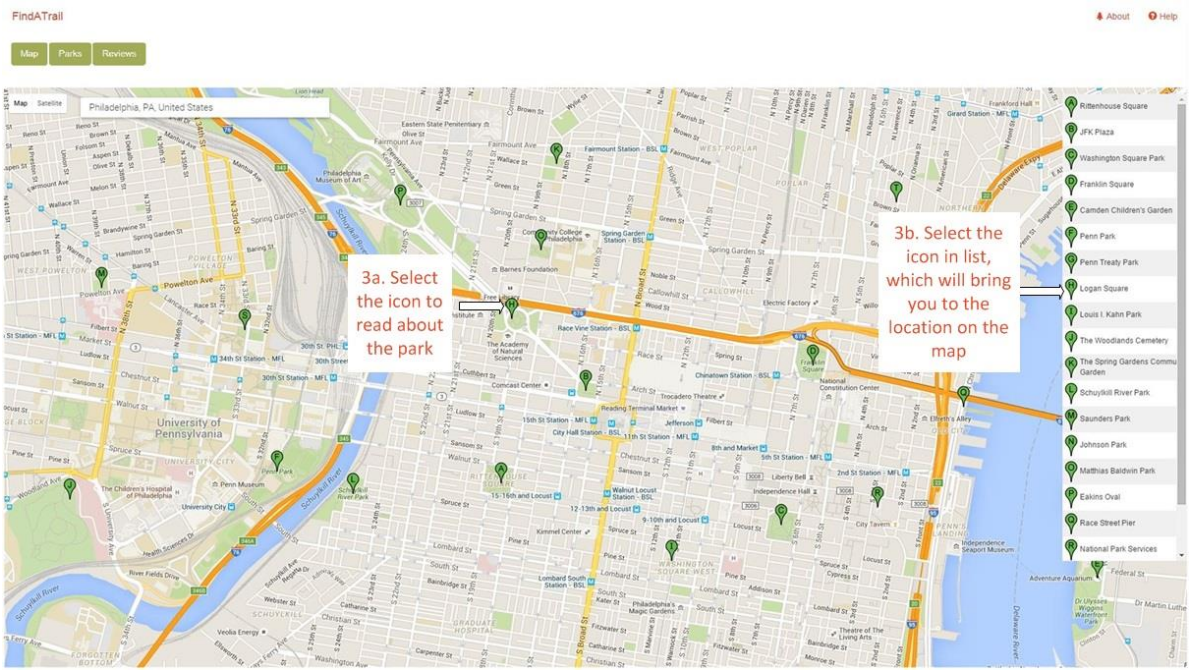
Home Page

This is the Home Page of FindAPark where the user can view 5 scenic backgrounds and either navigate to the Map Page or Search for a city.

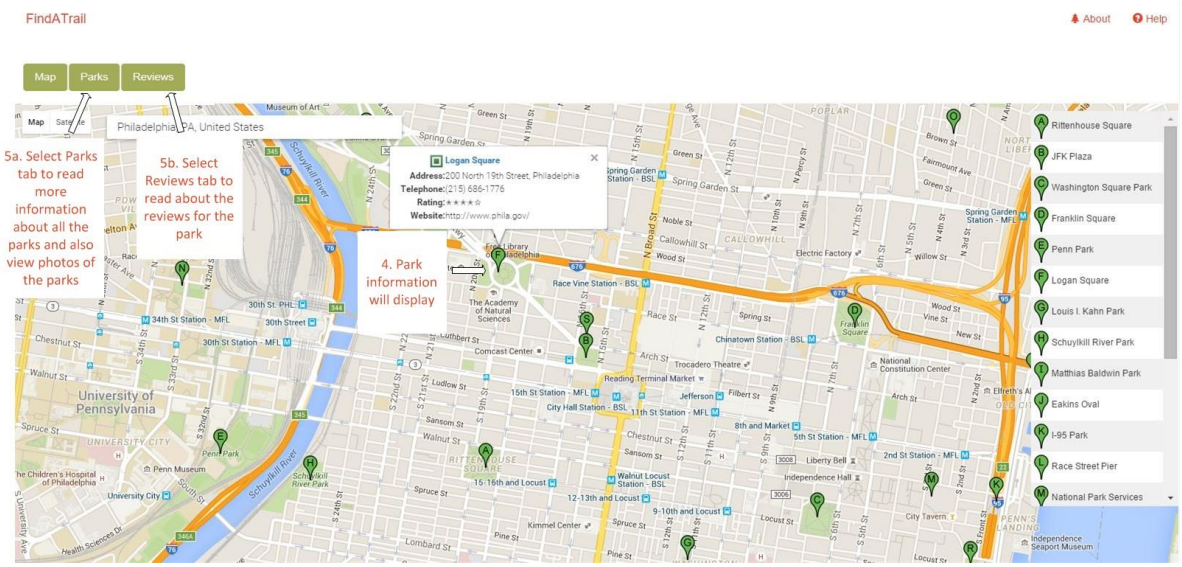
Map



This is the Map Page of FindAPark where the user can search for a city location and then select the park icon to read more about the park. In this first section, the user types in a city location into the search bar and then presses enter or selects an option in the dropdown.



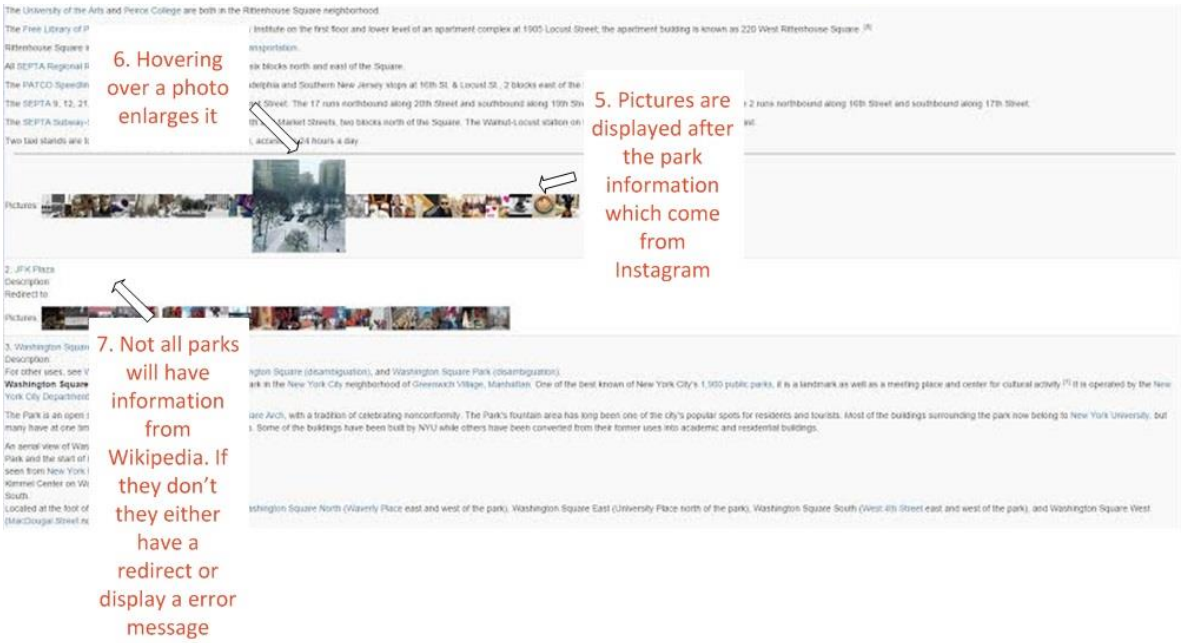
This is the second section of the Map Page where the user has already selected “Philadelphia, PA United States” as their city location. Now they can select a park icon, either from the right table or from the map.



This is the third section of the Map Page where a park icon was selected, “Logan Square”. Now the Google Plus information will display, if available (Address, Telephone, Rating, Website, and Name). Also, the user can either select the Parks button in the navigation to read more about the Parks displayed on the page or select the Reviews button in the navigation to read reviews about the Park (“Logan Square”) that they selected.

Parks

This is the Parks Page where the user can view photos from Instagram and read helpful information from Wikipedia about each park. This page displays all the parks that were listed on the Map Page. In this first section I state that once a city is select on the Map or Home Page they can then view information about the park in that area. The selected city's name will display on the top left corner, "Philadelphia, PA United States". The user can navigate to the Reviews Page by either clicking the Reviews navigation button or directly clicking the parks name.



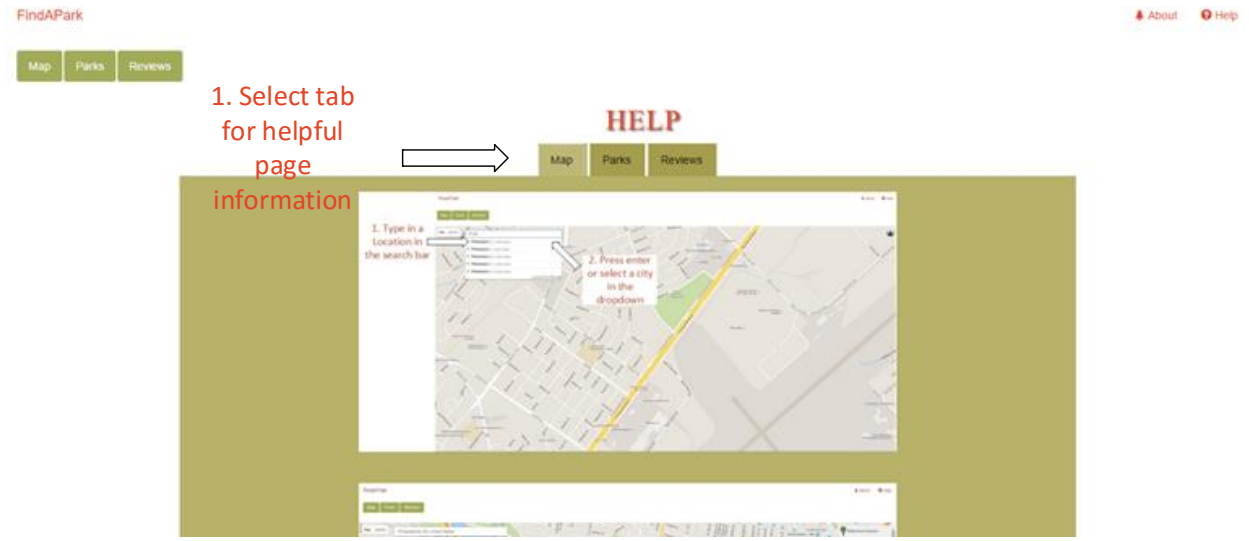
This is the third section of the Parks Page where the user has viewed the information provided by Wikipedia and now scrolls to the bottom of the row, where images from Instagram will be displayed. They have the option to hover over the photo, in order to enlarge them or click the photo to navigate to the actual photo on Instagram. If no information is found about the park, an error message is displayed.

Reviews

This is the Reviews Page where the user can read reviews about the park that they have selected from either the Map Page or Parks Page. In the section the city and park name is displayed in the top left corner of the page. The user can also view where the review came from and who wrote it by look under the text in the right and left corner of the row.

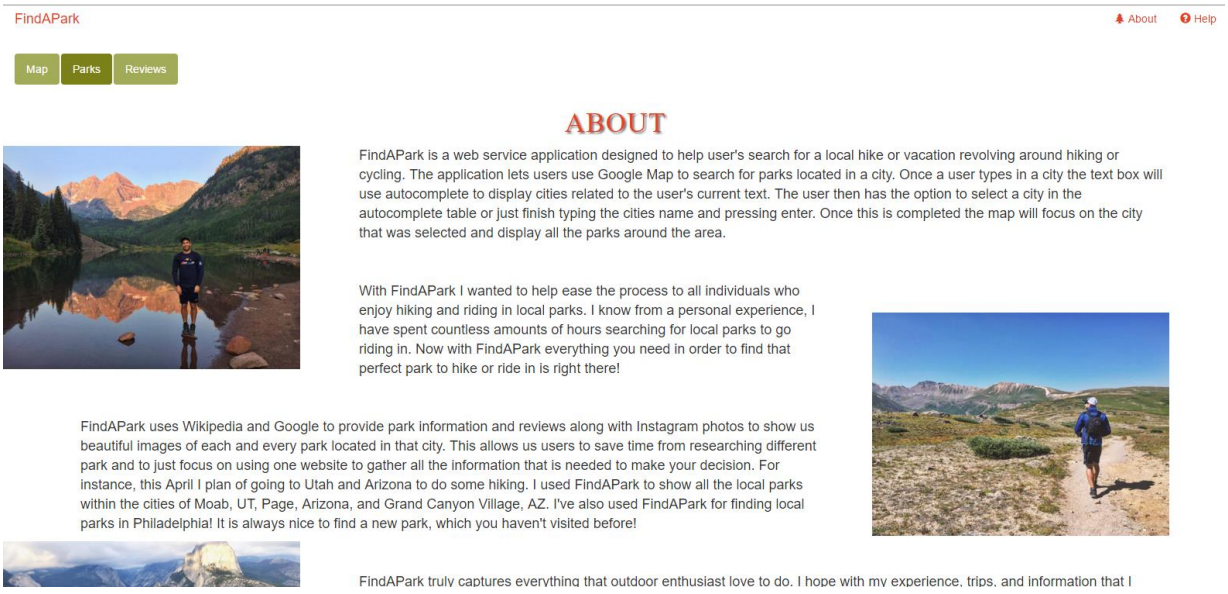
This is the second section of the Reviews Page where the user has selected the reviewers name from reviews table. Once the name is clicked, the user is then taken to their Google Plus Page.

Help



This is the Help Page of FindAPark where the user can view helpful information about each page. The user can select on the central navigation tabs in order to read helpful information about each of the main 3 pages.

About



This is the About Page of FindAPark where the user can read information about the developer.

Verification & Validation Report

1. **Events in Use Case descriptions should map to activities in the Activity Diagram**

- The events in UC: Home Page maps to the Home Page activity
- The events in UC: Map maps to the Map activity
- The events in UC: Parks maps to the Parks activity
- The events in UC: Reviews maps to the Reviews activity
- The events in UC: Help maps to the Help activity
- The events in UC: About maps to the About activity

2. **Object nodes in an activity diagram must be mentioned in Use Case descriptions**

- Object nodes in AD: Home Page is mentioned in the Home Page Use Case
- Object nodes in AD: Map is mentioned in the Map Use Case
- Object nodes in AD: Parks is mentioned in the Parks Use Case
- Object nodes in AD: Reviews is mentioned in the Reviews Use Case
- Object nodes in AD: Help is mentioned in the Help Use Case
- Object nodes in AD: About is mentioned in the About Use Case

3. **Sequential ordering within the Use Cases should match ordering in Activity Diagram**

- Sequential ordering of the use cases matches the Activity Diagram up to the Home Page node. At that point the user will have multiple options available to them and the ordering at that point is no longer important.

4. **There must be a one-to-one correspondence of Use Cases in the Use Case Diagram and Use Case descriptions.**

- The Home Page use case corresponds to the Home Page node in the Use Case Diagram
- The Map use case corresponds to the Map node in the Use Case Diagram
- The Parks use case corresponds to the Park node in the Use Case Diagram
- The Reviews use case corresponds to the Review node in the Use Case Diagram
- The Help use case corresponds to the Help node in the Use Case Diagram
- The About use case corresponds to the About node in the Use Case Diagram

5. **All actors listed in a use case description must be portrayed on the use-case diagram**

- User is portrayed in the use case diagram

6. **Include stakeholders listed in the use case description as actors in the use-case diagram**

- User is portrayed in the use case diagram

7. **All relationships listed in a use-case description must be portrayed on a use-case diagram**

- The association between Home Page and User is portrayed on the use case diagram
- The association between Map and User is portrayed on the use case diagram
- The association between Parks and User is portrayed on the use case diagram
- The association between Reviews and User is portrayed on the use case diagram
- The association between Help and User is portrayed on the use case diagram
- The association between About and User is portrayed on the use case diagram

Challenges Encounters

Originally the plan was to develop a web application for users to find different trails by searching for a park. Since Google maps doesn't allow users to specify the keyword trails, FindATrail had to be modified to FindAPark. This now allowed users to use the web application to find the keyword parks within a city. Even though this was a different approach from the originally plan, the application still works successfully. The user now has to input a city, for example Philadelphia. Once this is done, all the parks within Philadelphia will display on the map.

Another challenge that was encountered was building a web crawler/ web scraper. Due to copyright laws, it was difficult to use this method in order to pull park information from websites like Yelp, TripAdvisor, and Trails.com. After contacting these websites, a membership from a partnering company was needed in order to access their data for personal use. All three companies required a purchase of an API key (cost approximately $100) to use their content. If a web scraper was used to gather the information and pictures from their page, it was not legally acceptable. Also, a student API key was not offered.

Time management was another difficulty that was encountered. During my capstone, dealing with my job, another class, and coaching while trying to complete FindAPark. Overall, it was hard to balance the work load without burning out. Especially noted with the length of the capstone developing, the application got more difficult with the constant API updates that Google was performing.

Finally, a major challenge that was encountered was using the Home Page in FindAPark to pass the search city to the Map Page. With how Google Places API's functionality works, this wasn't allowed. FindAPark is able to center on the location of the city, but in order to display the park location icon, the user needs to select the cities name in the search bar.

Lessons Learned

Developing FindAPark taught integration of multiple APIs. Working with Google Places, Wikipedia, and Instagram all three APIs offered a different feature to the web application. Instagram used Instafeed.js which is javascript that adds Instagram photos to a website, by passing in feed parameters such as a tagName. For instance looking up a tag name of Philadelphia, the developer would do a get: 'tagged', tageName: 'Philadelphia', and then the clientId: 'Unique ID' (This is the unique

identifier that you are provided from your Instagram page). For the Wikipedia API, the developer just needs to do run a getJSON command that points to Wikipedia with the page equaled to the search. For instance, FindAPark uses the park, so Rittenhouse Square. Once this call is made, the getJSON call retrieves the page information and then displays in all in a specified div. CSS was used to clean up the Wikipedia page and only display necessary information. The final API that was used and was the main focal point of FindAPark was the Google Maps/ Places API. This API provided the functionality for both the Map and Home Page. It also controlled the autocomplete functionality that is used on both the Home and Map Page. This API allows the developer to use a wide variety of features, for FindAPark the map, autocomplete, search, satellite, and info table were all used. These features all improved the application and made it useable for all ages.

A few other lessons that were learned by developing FindAPark was how it is critical to be open-minded. When developing an application that is using all new technology features, one needs to understand that all APIs might not work together. Because of this, one needs to be open with a trial and error approach and learn to accept that since these APIs might not work together now, by providing feedback to the developers of the API can improve their product and help you complete your project in the future. Several days were spent trying to find a web crawler/ scraper API that could access the data that was needed for this application, but after a few weeks of testing the Wikipedia API was a much easier and safer approach to use due to it being open source.

Suggestions for Additional Research

FindAPark currently uses an Instagram and Wikipedia plugin in order to pull pictures and information from these websites. These two websites aren't the only resources that can be used to generate photos and information about each park. With special memberships such as, Yelp and TripAdvisor, the developer can use these APIs to generate more reviews, pictures, and information about each park. By adding a new photo, this could allow the users more of an idea of what the park has to offer. Using Google images could also be added to the parks page to increase the amount of photos that are offered by FindAPark.

Another idea for Additional Research is to reach out to Google and ask them to add more keywords for their API. By doing this you can expand the search results so both parks and possibly trails within the parks are displayed. If Google doesn't add the keyword for trails then another option would be to contact National Park Services or even local parks to see if you can pull there trail information to add to FindAPark.

Finally, another feature that can be added is to allow users of FindAPark to review the parks from within the web application. There might be a permission related issue with Google Plus, but this is something that should be looked into and could improve the web application. Currently, all the reviews are based off of what people have said in Google Plus, so this feature would post it to Google as well as FindAPark.

Version Control

Date	Version	Name/Author	Description of Changes/Review	Full Document Name
10/28/15	1.0	David Laratta	Original Author	CISCapstone.docx
2/14/16	1.1	David Laratta	Original Author	CISCapstone.docx
4/17/16	1.2	David Laratta	Original Author	CISCapstone.docx